Systems-of-Systems Programmatics: Guidelines for Program Managers

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213

Jim Smith, Craig Meyers, David Fisher

Presentation for Systems and Software Technology Conference 2007

June 21, 2007

Software Engineering Institute | Carnegie Mellon

Purpose

To introduce a set of guidelines for dealing with systems of systems programmatics, and show how these can aid in the identification and resolution of issues.

Agenda

Background

Guidelines for the program manager: the “laws of programmatics”

Discussion

Conclusions

Future directions

DOCTOR FUN

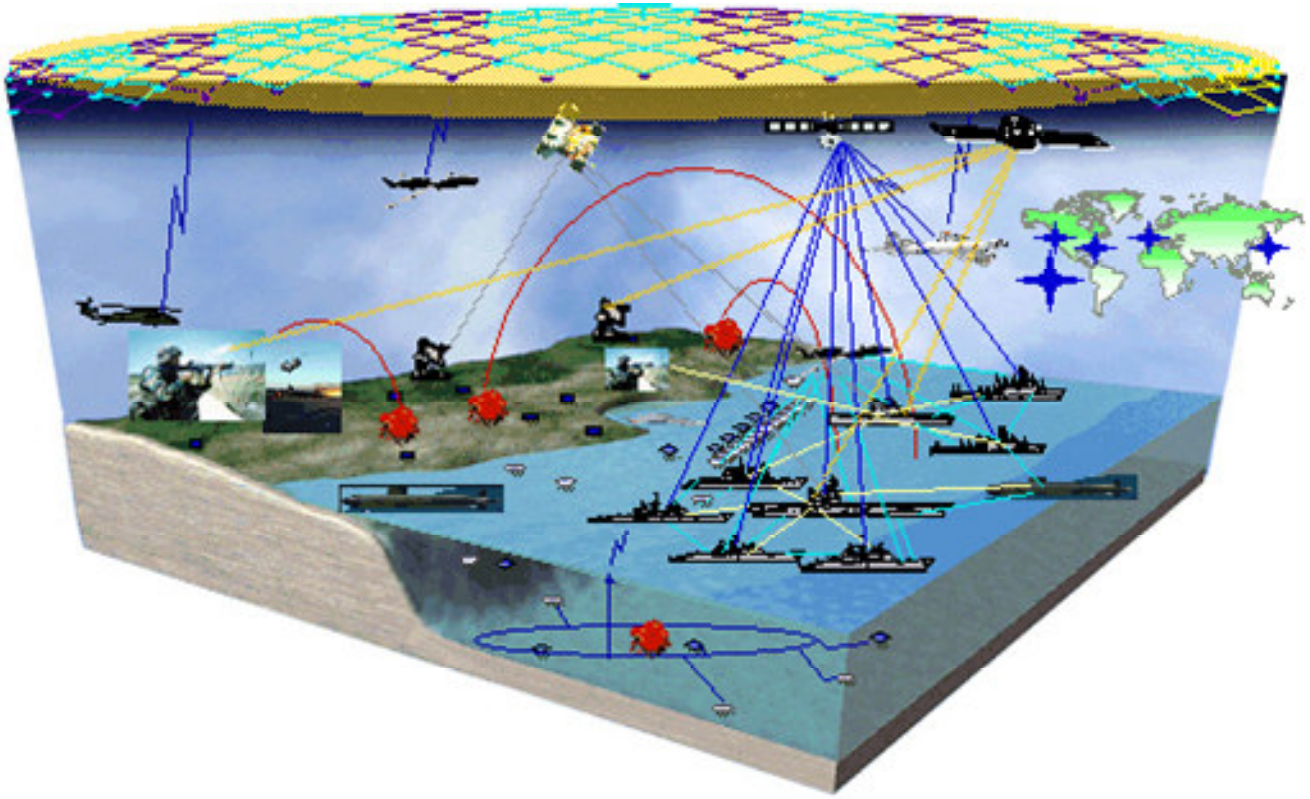
Oct 2002

The daydreams of cat herders

Used by permission of the artist

Background$_1$

What is the vision for net-centric operations?

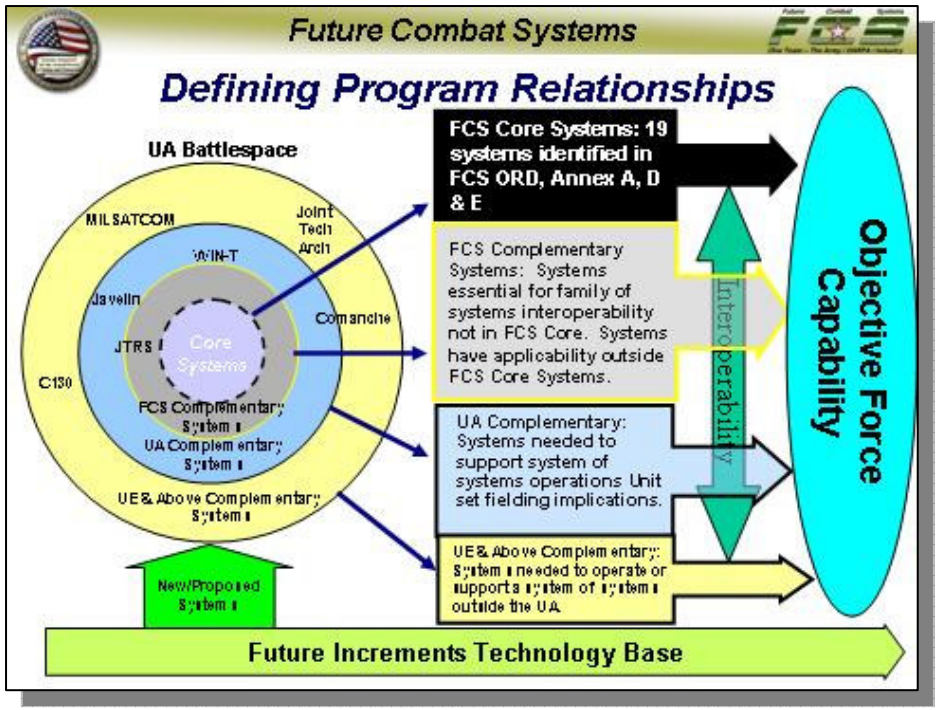


Bottom line: Seamless, ubiquitous interoperability enabling dynamically-composable capabilities to achieve desired mission effects

Background$_2$

Current approaches to providing capability based on integration of and interoperation between systems

- Constituent systems not necessarily developed with "broad" interoperability in mind
- Acquisition (development, sustainment, etc.) focused on their individual systems/programs, with their individual requirements, funding, users, etc.
- Integration "after the fact"



Example from "Future Combat Systems Program Relationships"

Background$_3$

Results in disconnects between the goals and reality:

Goal

- Seamless interoperability
- Dynamic, flexible, responsive to changing environment and threat, survivable
- Programs will “cooperate and graduate”

Reality

- Interoperability either insufficiently defined (e.g., “interoperable with *XYZ* system”) or too narrowly constrained (e.g., “interoperable with *XYZ* system, using TDMA over 25 KHz UHF DAMA STACOM …” etc.
- Integration results in brittle systems that are fragile and difficult to sustain
- Stovepiped programs/systems not incentivized to engage in altruistic behavior

Background$_4$

For many reasons, systems of systems are currently the preferred approach to providing net-centric capabilities

Certain inescapable consequences of systems of systems:

- Autonomous constituents with independent operations and management
  - Includes people, organizations, software agents, etc.
  - Source of independent actions and decisions
- Independent operation/evolution of each constituent
  - Can respond to new technology and mission needs at its own pace and direction
- Emergent behavior
  - "Whole is different than the sum of the parts"
  - Indirect and cumulative effects of influences, actions, interactions

These require a fundamentally new approach to understanding and managing the programmatics (i.e., the interrelationships between issues/decisions in management, development, and operations)

Background$_5$

Some critical aspects of desirable systems of systems programmatics include:

- Collective, collaborative behavior to accomplish systems of systems goals
- Understanding the interrelationships and interdependencies between management, development, and operational domains
- Distributed execution authority
- Effective decision making in the presence of inconsistent, incomplete, and incorrect information

These often conflict with the "system-centric" pressures on a program manager to satisfy their instant cost, schedule, and performance requirements

What is needed are some guidelines to help program managers in the execution of their responsibilities consistent with systems of systems principles: the "*Laws of Programmatics*"

The "Laws of Programmatics"

Zeroth Law

An actor or constituent (hereinafter referred-to as *actor*) shall not take unilateral action within their program to the detriment a system of systems, or through inaction, knowingly allow detriment to come to a system of systems

First Law

An actor shall not take unilateral action within their program to the detriment of another program, or through inaction, knowingly allow detriment to come to another program.

Second Law

An actor shall comply with all applicable laws, regulations, directives, policies, etc. issued by competent authority except when doing so would conflict with the zeroeth or first laws.

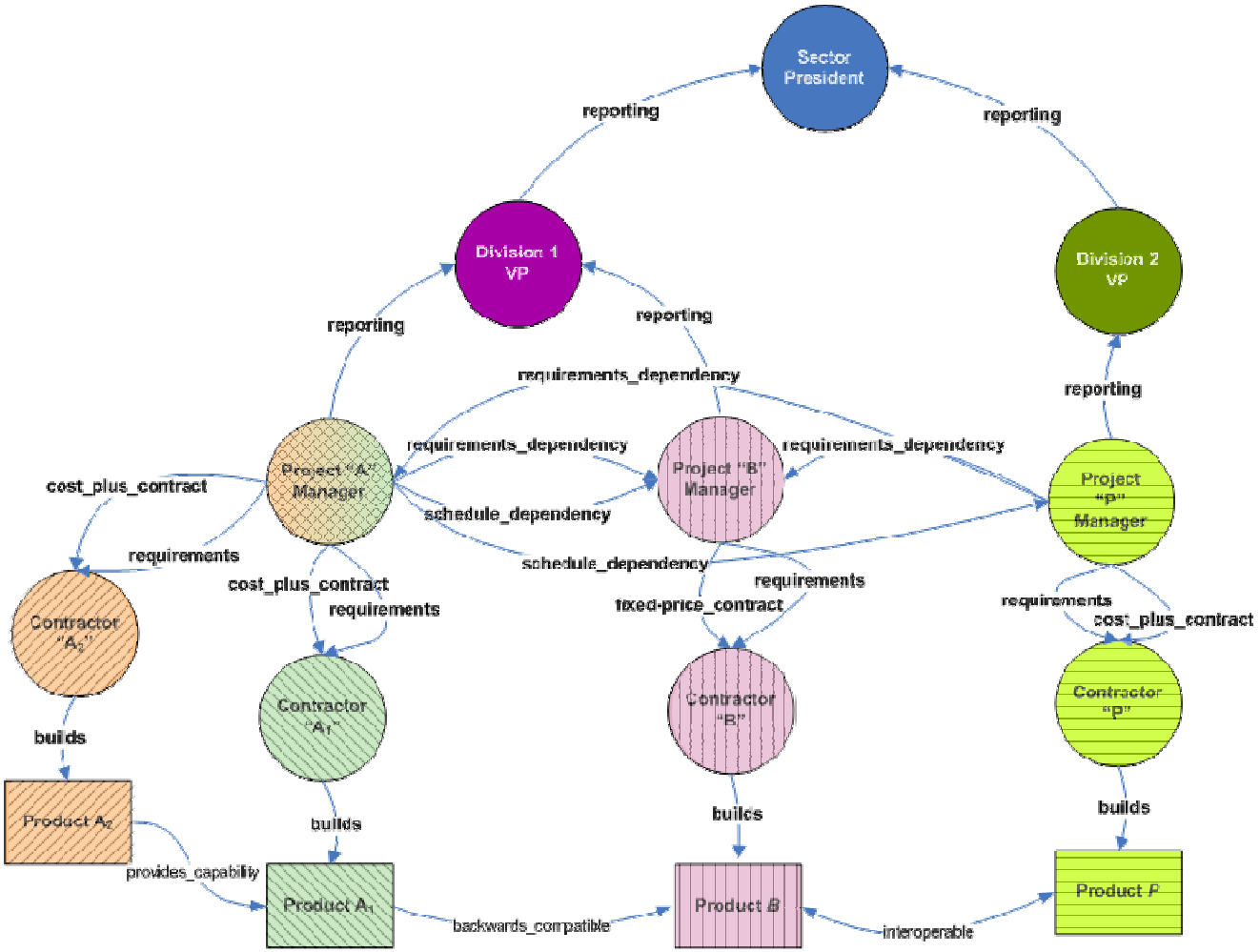
Third Law

An actor shall take corrective actions to remedy any risks, issues, problems etc. in their program except when such actions conflict with any higher law.

Fourth Law

An actor may "be creative" as long as their actions do not conflict with any higher law.

The Laws in Action



Zeroth Law

An actor shall not take unilateral action within their program to the detriment a system of systems, or through inaction, knowingly allow detriment to come to a system of systems

Applies when it is known (or can reasonably be inferred) that a program is part of one or more systems of systems

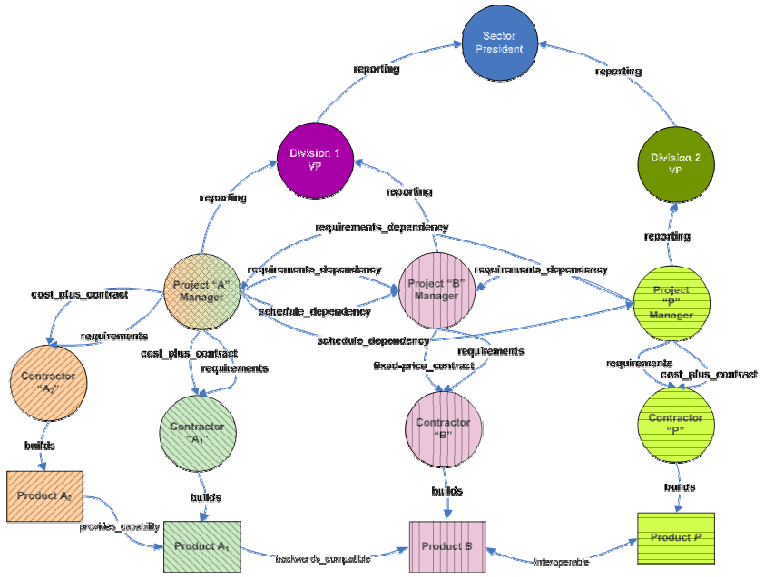
Obligates a program to not take action to the detriment of the system(s) of systems in the pursuit of its goals. For example:

- Capability "X_4" is critical for the system of systems, but not for the success of program "*X*"
- Program "*X*" should not, therefore, decide to delete capability "X_4" on its own, even when it has the authority to do so under existing acquisition program guidance, etc.

Zeroth Law

To comply with the zeroth law, several actions need to be undertaken:

- Program "*P*" must explore options to prolong its operation in the event that system "A_1" cannot be ready on time
- Before any additional changes to systems "*P*" and "*B*," program "*A*" must be allowed to weigh in on the decision, and suggest possible alternatives that won't further endanger their ability to meet schedule
- Program "*A*" must reconsider their risk reduction effort (system "A_2") if it will be unable to support the system of systems interoperability requirements.



First Law

An actor shall not take unilateral action within their program to the detriment of another program, or through inaction, knowingly allow detriment to come to another program.

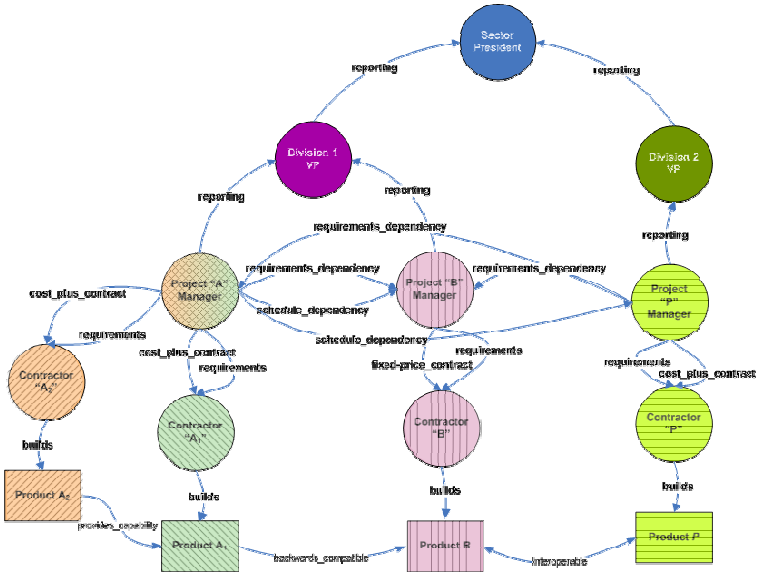
Obligates a program to not take action to the detriment of another program. For example:

- Program "*Y*" has an interoperability requirement with program "*X*"; this interoperability is critical to the success of program "*X*," but not for program "*Y*"
- Program "*Y*" should not remove or modify this requirement on its own

First Law

Independent program actions causing harm to other programs:

- Changes to system *P* require corresponding changes to system *B*
  - Impacts design/development for system A_1
- System *A* risk-reduction effort (A_2) will impose significant changes on system *B* if implemented



Second Law

An actor shall comply with all applicable laws, regulations, directives, policies, etc. issued by competent authority except when doing so would conflict with the zeroth or first laws.

This obligates a program to follow the applicable laws, directives, etc. unless doing so would harm another program or the system(s) of systems

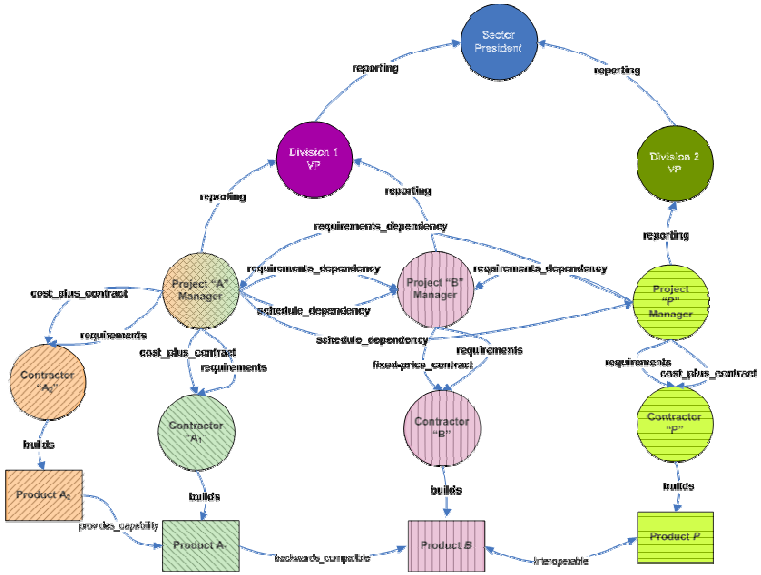
One consequence of this is to require that a program "throw an exception" when there are conflicts between the laws, directives, etc. and the needs of another program, or the system(s) of systems. For example:

- Funding for program "*X*" is reduced, which will result in a delay to fielding a capability that is critical to the system of systems
- Program "*X*" should identify the conflict between its goals and those of the system of systems, and attempt to find a solution (or force reconsideration)

Second Law

Conflict between goals, directives, etc. of individual programs and the system of systems:

- Planned retirement of system *P* jeopardizes system of systems
  - What if system A_1 not ready?
  - What if program *A* risk reduction effort (system A_2) is implemented?



Third Law

An actor shall take corrective actions to remedy any risks, issues, problems etc. in their program except when such actions conflict with any higher law.

This obligates a program to not take actions to the detriment of the system(s) of systems or other programs, or violate applicable laws, directives, etc. while mitigating their own problems

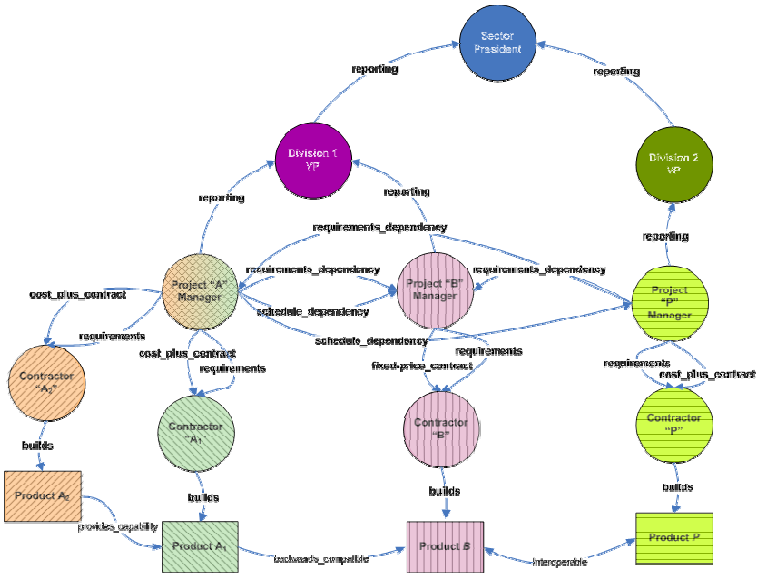
As an example:

- Program "*Y*" has a backwards compatibility requirement with program "*X*," and is experiencing significant cost and schedule pressures that can be ameliorated by deferring this capability
- Program "*Y*" should not defer this capability on its own

Third Law

Conflict between risk reduction efforts or mitigation strategies of individual programs versus systems of systems capabilities:

- System A_1 risk reduction effort (A_2) satisfies system A_1 requirements, but adversely impacts system *B* and broader system of systems requirements
- Is there a minimally-acceptable subset of interoperability that system A_2 could provide that would preserve system of systems goals?



Fourth Law

An actor may "be creative" as long as their actions do not conflict with any higher law.

While in the context of a single program, there may be numerous examples of effective solutions to questions about organizational issues, risk management, budgeting, etc.,

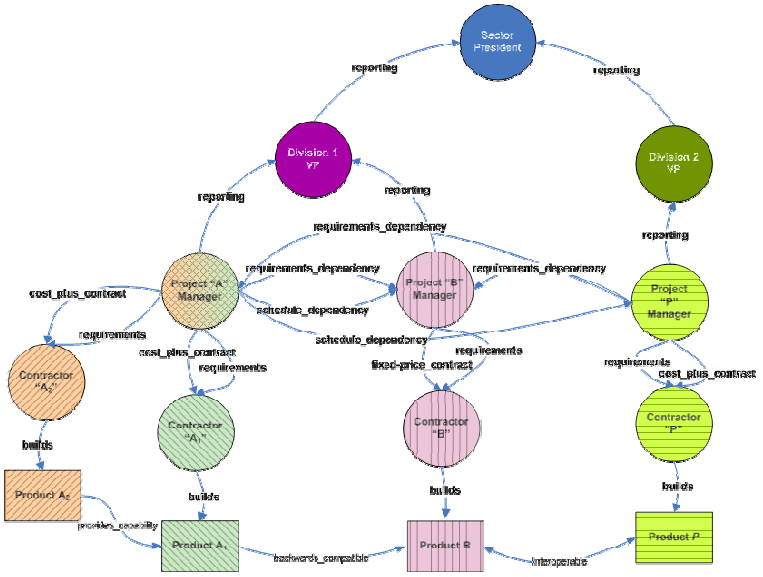
The same is not true for systems of system: most of what program managers have learned through their experiences and training leave them ill-prepared to deal with the uncertainties of systems of systems

- Since there are no "off-the-shelf" solutions, you should explore some "out of the box" possibilities

Fourth Law

When there are no known “tried and true” solutions, then try something else!

- Keeping your “head down,” or your “nose to the grindstone” won’t make these problems go away
- Programs must consider the impacts of their decisions on every other program—and on the system of systems
- Programs need consider this unpleasant truth: sometimes the right answer is “no”



Conclusions

Systems of systems are the preferred approach to provide net-centric capability

Managing a program—and maintaining cost, schedule, and performance—has never been easy, and systems of systems bring many new challenges

- Conflicts between system-centric and systems of systems perspectives

A way to identify these conflicts, and provide program managers with guidance on how to proceed, would help

- The “laws” are an attempt to provide this

Future Directions

As interesting as the natural language representation of the laws may be, a formal definition is required

- Currently, working on a formal definition that integrates Deontic logics with precedent-based reasoning
- Desire a method for analyzing—and recommending courses of action—for conflicting normative obligations

Part of a long-term effort to be able to identify what organizational architecture(s) work better than others, under what conditions

- Goal is to be able to model the eigenbehaviors of organizations, defined by their "genomes," operating within a framework defined by the laws, when confronted by various systems of systems issues

Some Recent Technical Reports

Smith, J.; & Phillips, D. "*Interoperable Acquisition for Systems of Systems: The Challenges*" (CMU/SEI-2006-TN-034)

Brownsword, L.; *et al.* "*System-of-Systems Navigator: An Approach for Managing System-of-Systems Interoperability*" (CMU/SEI-2006-TN-019)

Fisher, D. "*An Emergent Perspective on Interoperation in Systems of Systems*" (CMU/SEI-2006-TR-003)

Meyers, C.; Smith, J.; *et al.* "*Requirements Management in a System of Systems Context: A Workshop*" (CMU/SEI-2006-TN-015)

Smith, J.; & Meyers, C. "*Exploring Programmatic Interoperability: Army Future Force Workshop"* (CMU/SEI-2005-TN-042)

Meyers, C.; Monarch, I.; Levine, L.; & Smith, J. "*Including Interoperability in the Acquisition Process"* (CMU/SEI-2005-TR-004)

Morris, E.; Levine, L.; *et al.* "*Systems of Systems Interoperability"* (CMU/SEI-2004-TR-004)

Contact Information

Jim Smith

(703) 908-8221

jds@sei.cmu.edu

http://www.sei.cmu.edu/staff/jds/

Craig Meyers

(412) 268-6523

bcm@sei.cmu.edu

David Fisher

(412) 268-7703

dfisher@sei.cmu.edu

ISIS Initiative

http://www.sei.cmu.edu/isis/isis-main.html